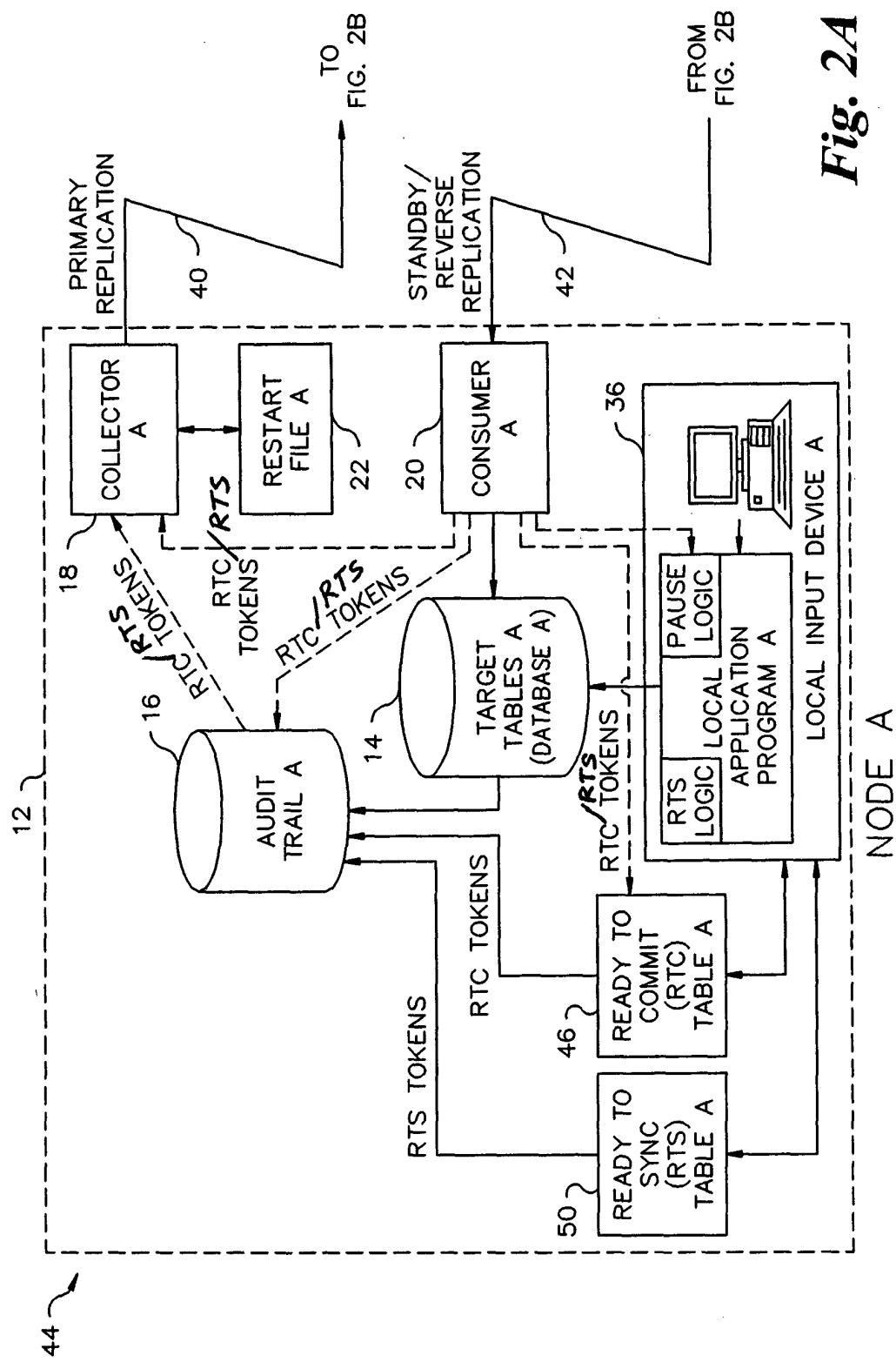


Fig. 1
(Prior Art)



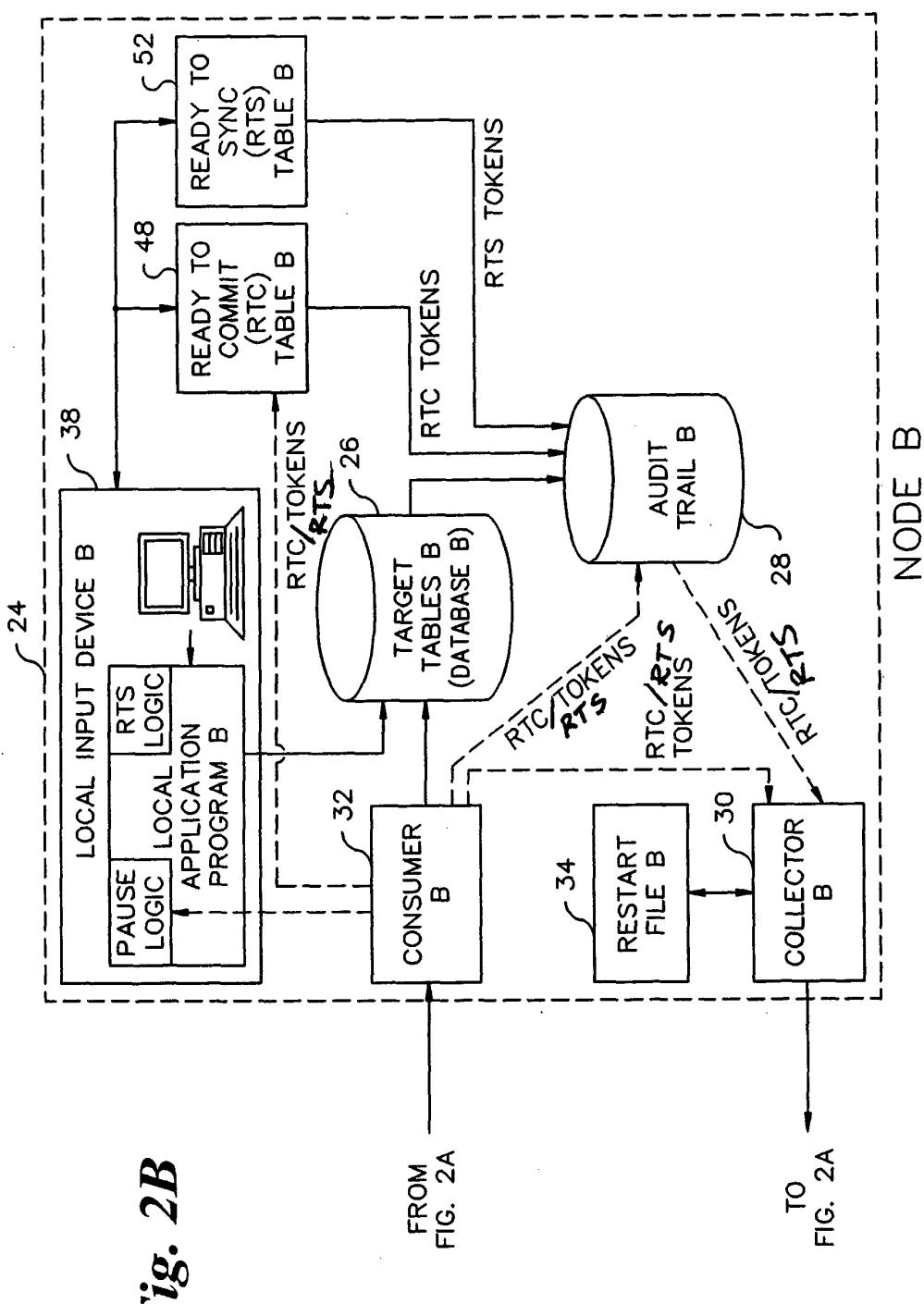


Fig. 2B

RTC TABLE A			AUDIT TRAIL A			AUDIT TRAIL B		
TIME	CONTENTS	TIME	HEADER	DATA	TIME	HEADER	DATA	
TRAN ID	FLAG	TRAN ID	TRAN ID	TABLE	TRAN ID	TRAN ID	TABLE	
t ₁		t ₁	101	BEGIN TRANS. 101	t ₁ + α	101	BEGIN TRANS. 101	
t ₂		t ₂	101	ACCOUNTS SMITH, JOHN, DEBIT \$10	t ₂ + α	101	ACCOUNTS SMITH, JOHN, DEBIT \$10	
t ₃		t ₃	101	ACCOUNTS DOE, JANE, CREDIT \$10	t ₃ + α	101	ACCOUNTS DOE, JANE, CREDIT \$10	
t ₄	101	0	t ₄	RTC TOKEN 101	t ₄ + α	101	RTC TOKEN 101	
t ₅	101	1	t ₅		t ₅ + α	101		
t ₆	101	*	t ₆	101	t ₆ + α	101	COMMIT TRANS. 101	
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	
t _n								

* NO ENTRY
 (TRAN ID 101
 HAS BEEN
 DELETED
 FROM TABLE)

Fig. 3

Replication Engine

44

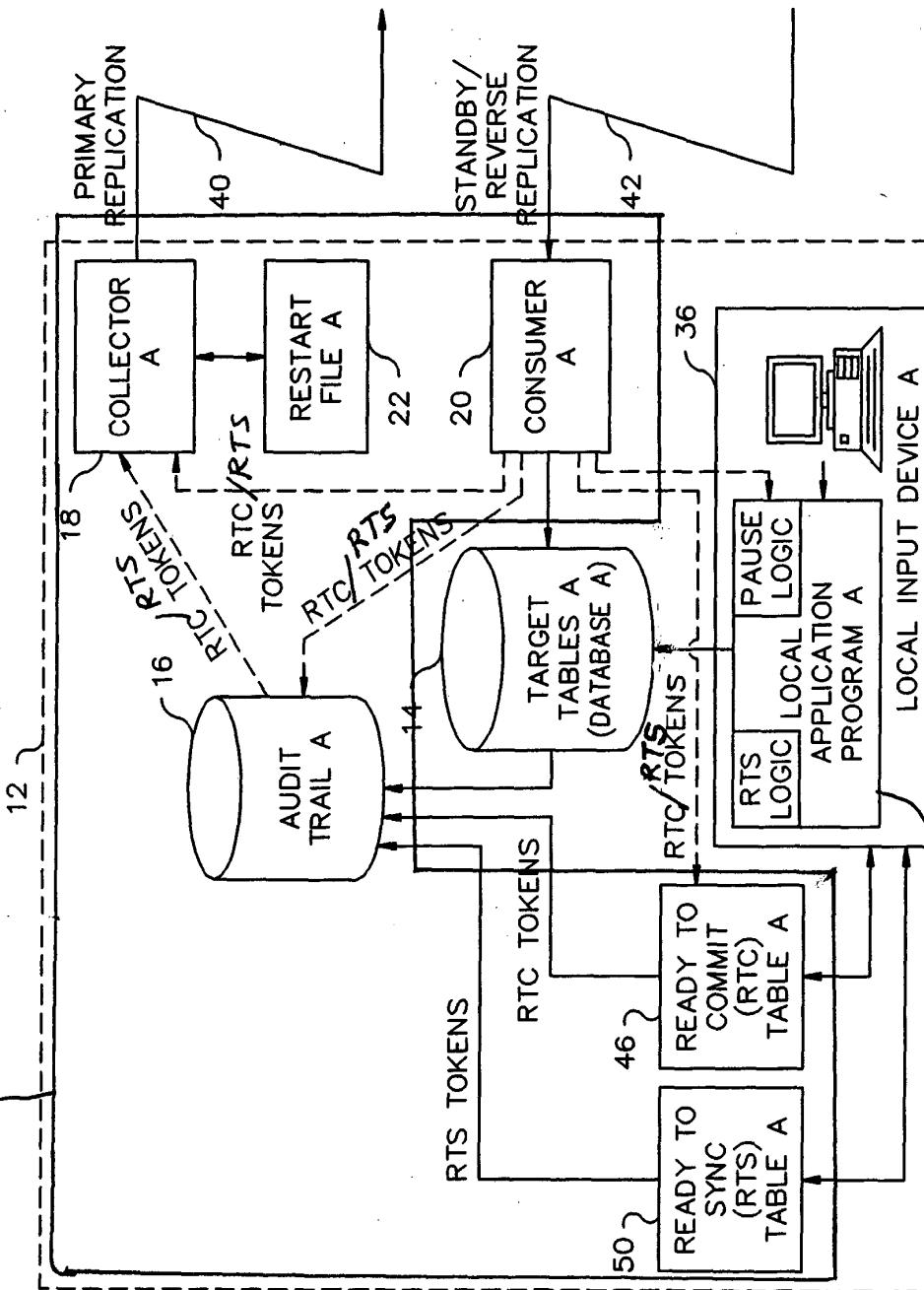
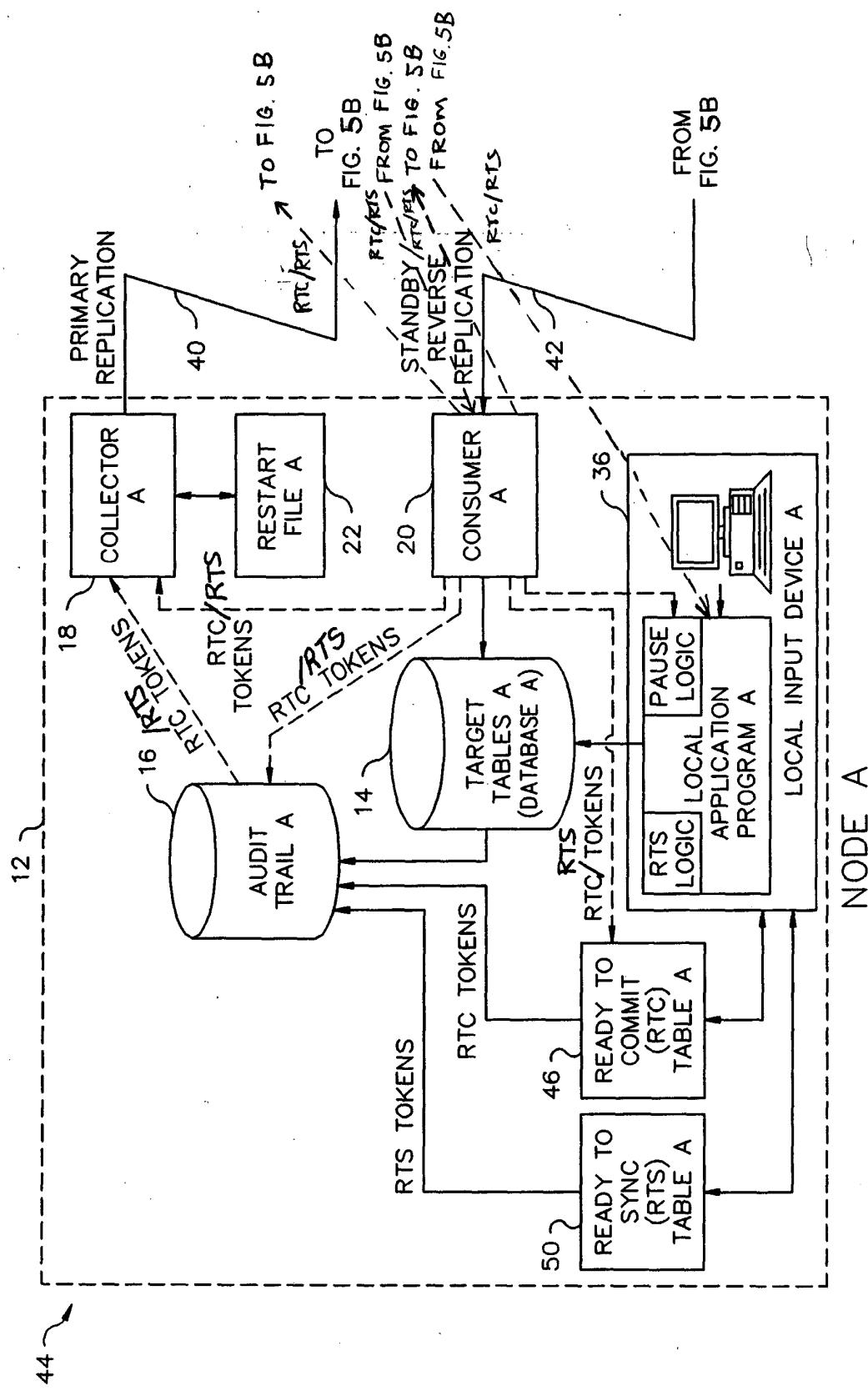
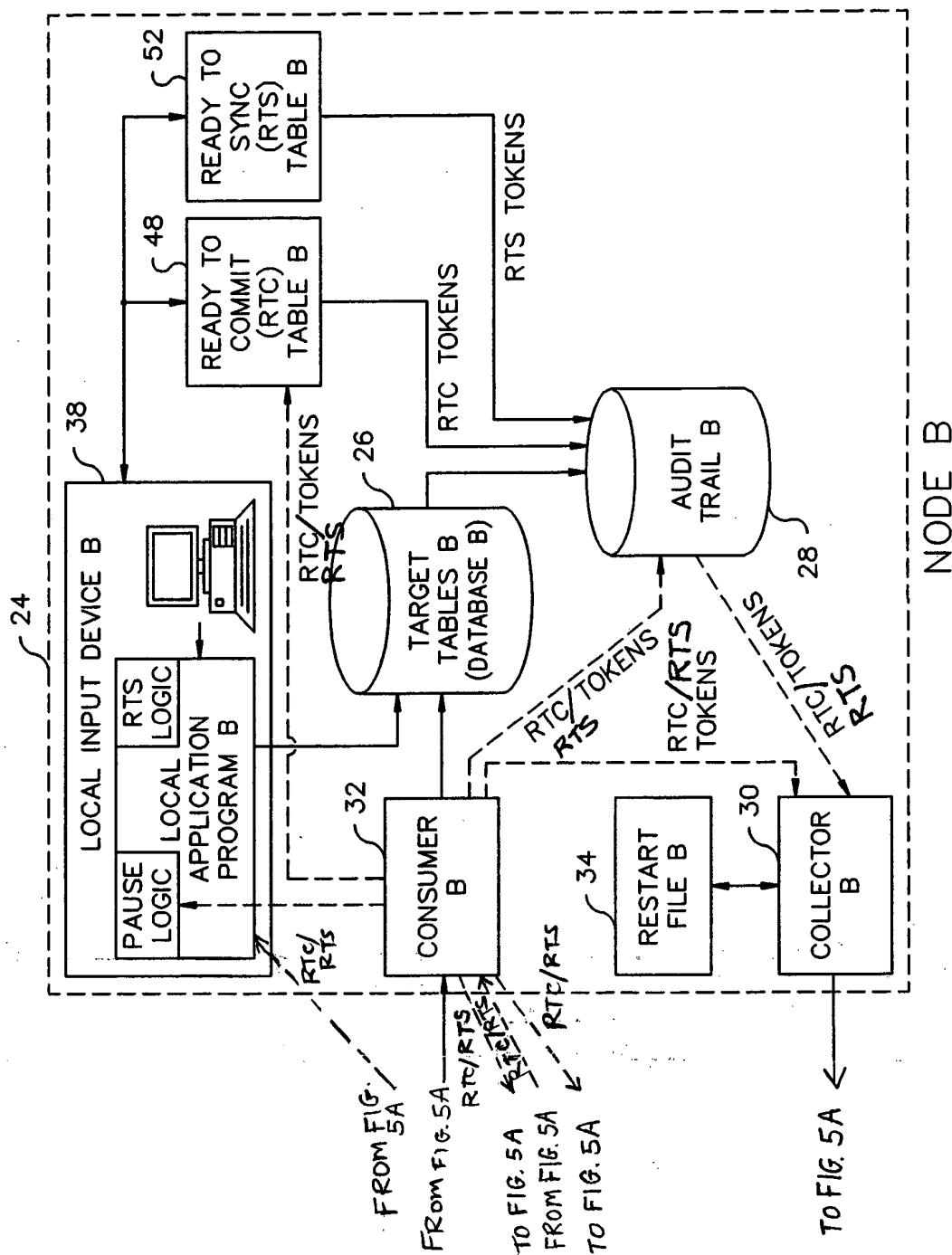


FIG. 4
NODE A

APPLICATION ENGINE





56
—
11

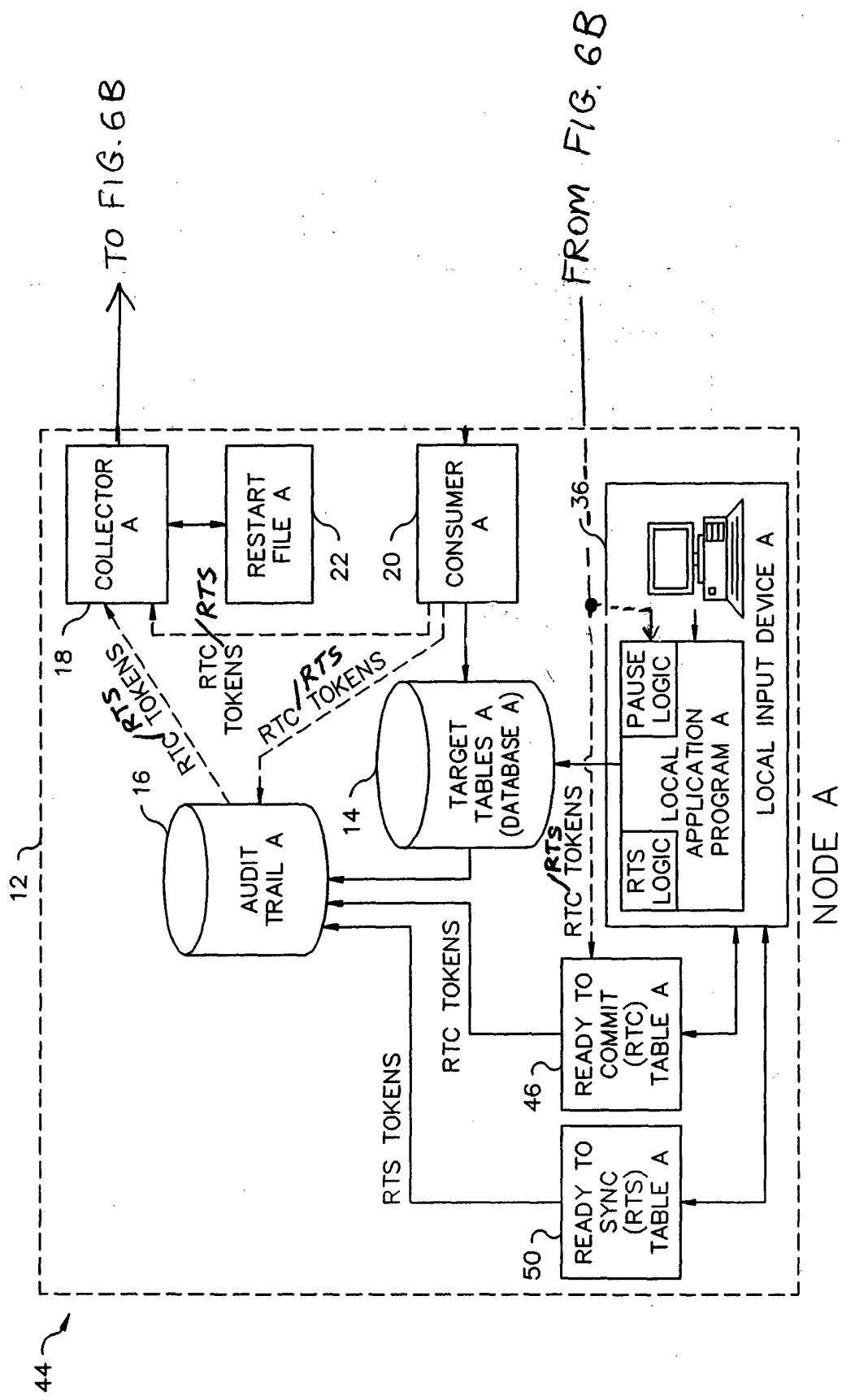


FIG. 6A

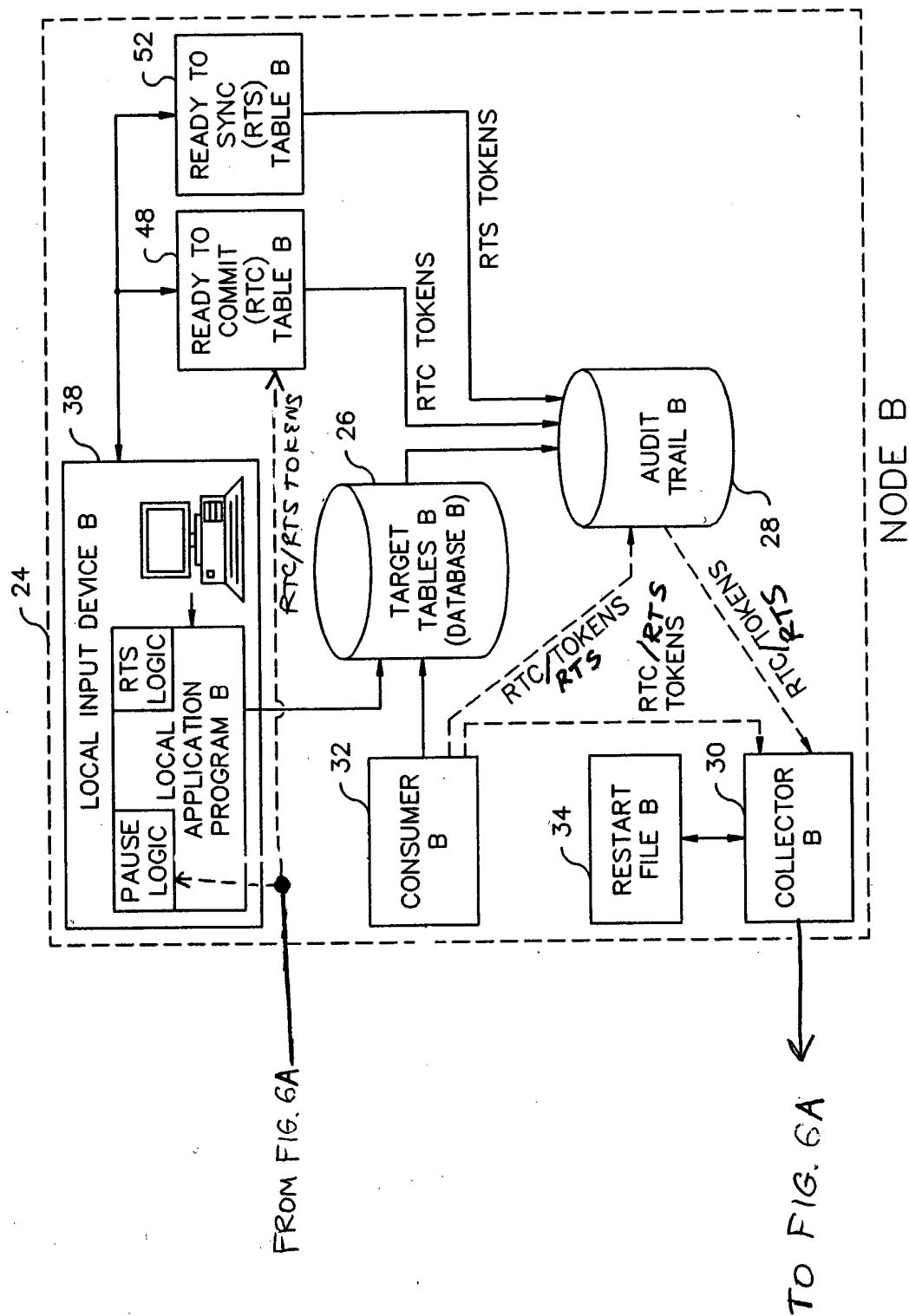


FIG. 6B

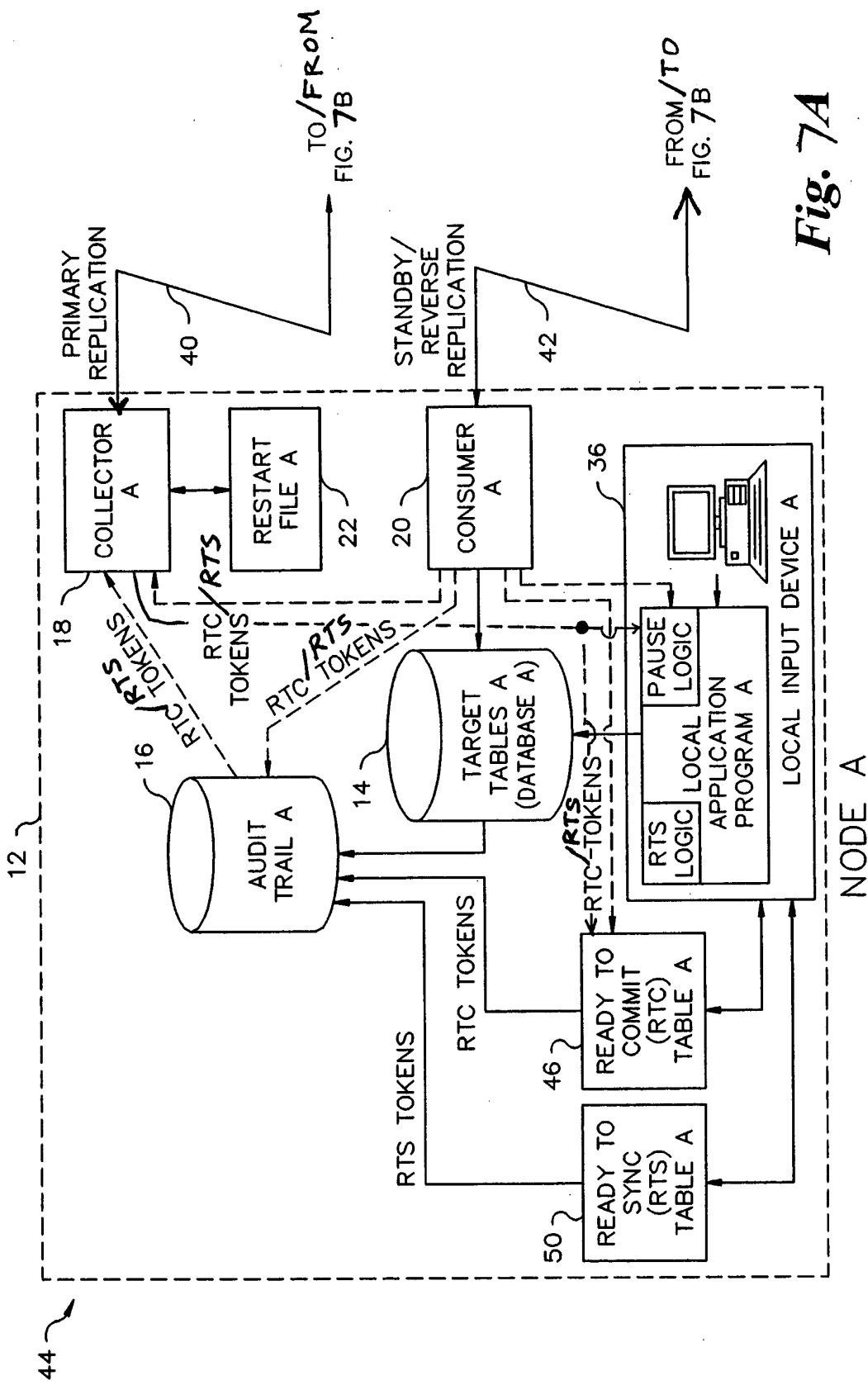
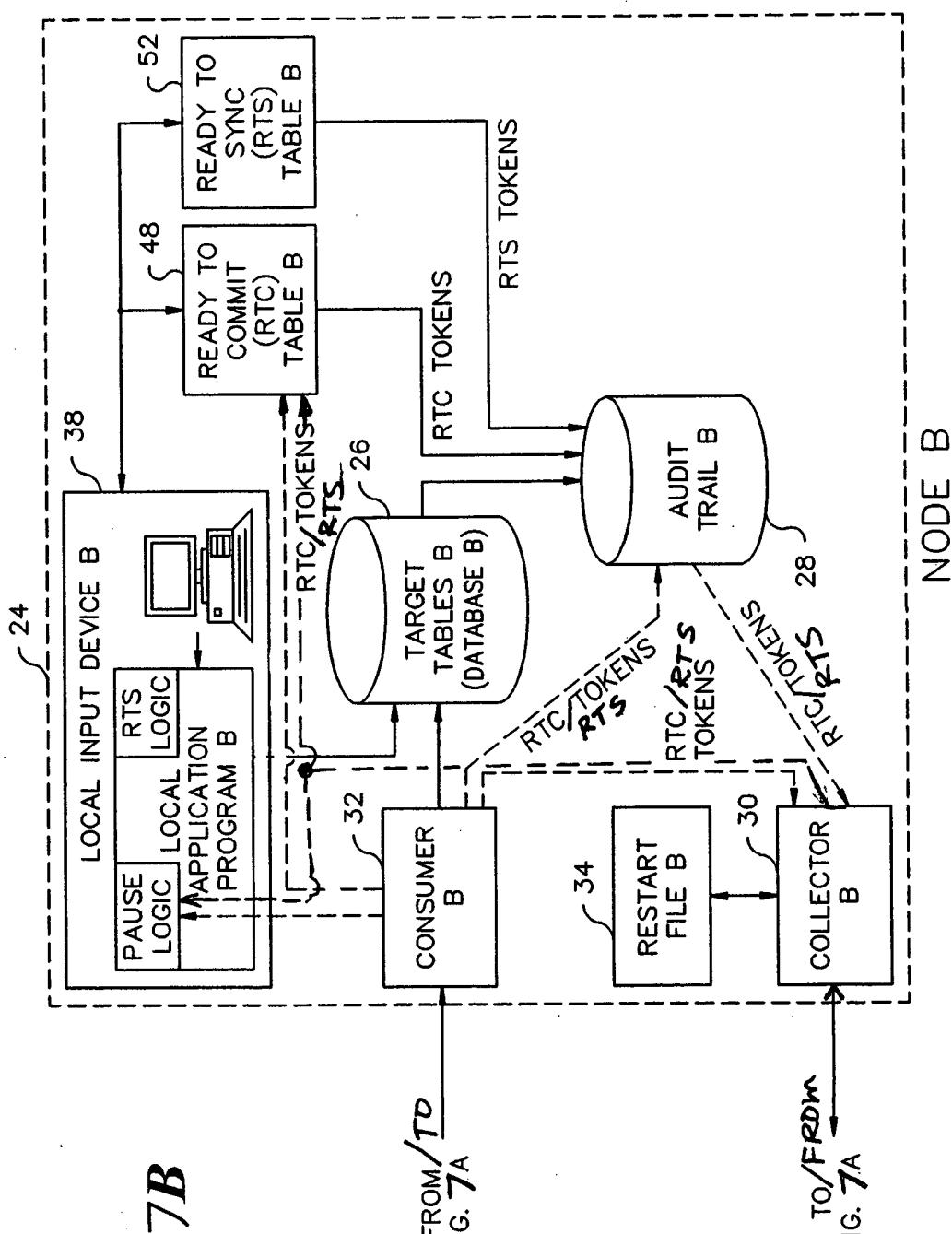
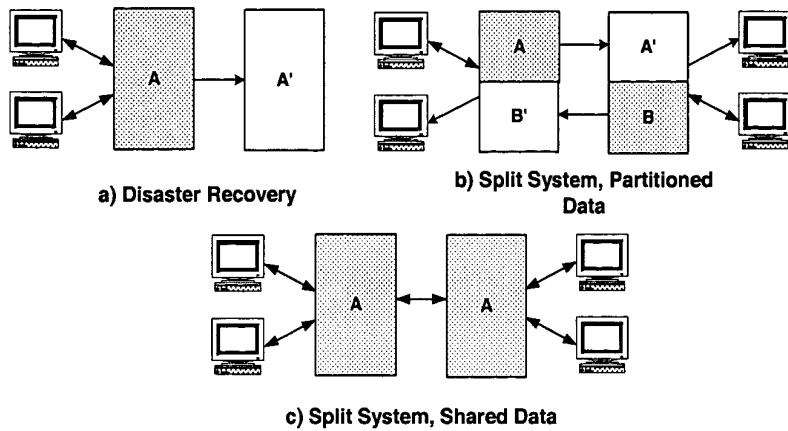


Fig. 7A



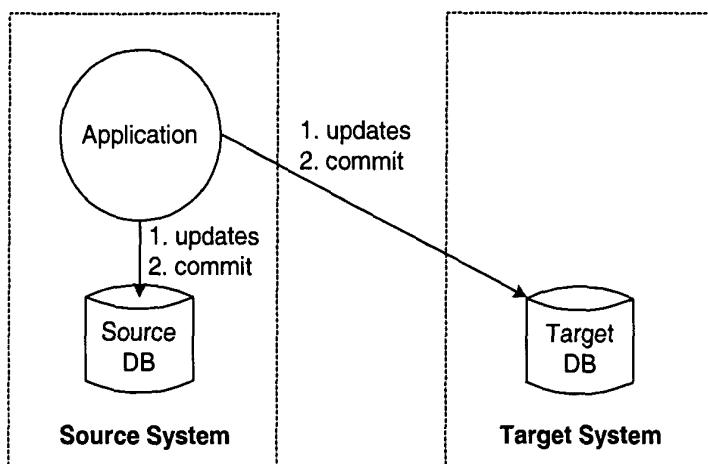
Node	Time	Transaction amount	Comments
A, B	T1		Account balance initially at \$1000, both nodes A and B synchronized.
A, B	T2	+\$200	Customer deposits (adds) \$200 to account balance from node A. Using the synchronous replication method, both nodes are updated to a balance of \$1200
A, B	T3	-\$75	Customer withdrawals (subtracts) \$75 from the account balance from node B. Using the synchronous replication method, both nodes are updated to a balance of \$1125
A, B	T4	-\$125	Customer transfers (subtracts) \$125 from the account balance to a foreign account from node A. Using the synchronous replication method, both nodes are updated to a balance of \$1000
	T5		All communications between nodes A and B is lost. A = B = \$1000
A	T6	+\$100	Customer adds \$100 to account balance on node A. A balance = \$1100, B balance = \$1000
B	T7	-\$50	Customer withdrawals \$50 from account balance on node B. A node balance = \$1100, B node balance = \$950.
A, B	T8		Full communications is restored, and the system resolves collisions as follows: A replays the transaction delta changes into B, B replays the transaction delta changes into A. The order of replay could be arbitrary. One could even 'merge' the two replay streams, using timestamps or sequence numbers, etc. The main point is that the account balances will match after all are replayed into the others.
A -> B	T9	+100	More specifically, replaying the A transactions to the B database adds \$100 to the B balance of \$950, making it \$1050. NOTE – we could be done by simply overwriting the A database balance with the B database balance, however replaying in the reverse provides a 'check' of the approach.
B -> A	T10	-50	Replaying the B transactions to the A database subtracts \$50 from the A balance of \$1100, making it \$1050.
A, B	T11		Compare the two balances, A=B and the databases are in sync. If other criteria are met, synchronous replication can be resumed.

Figure 8



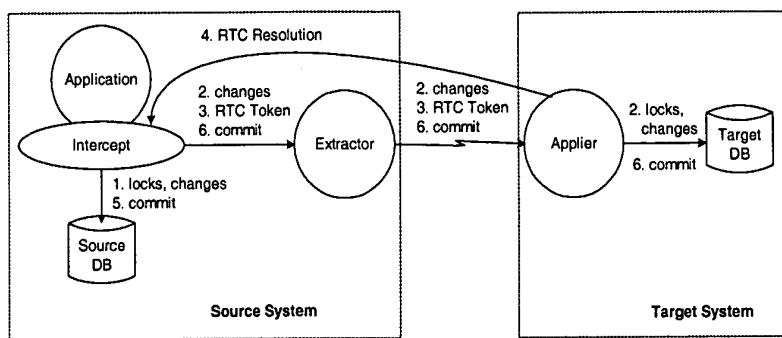
Split System Architectures

Fig. 9



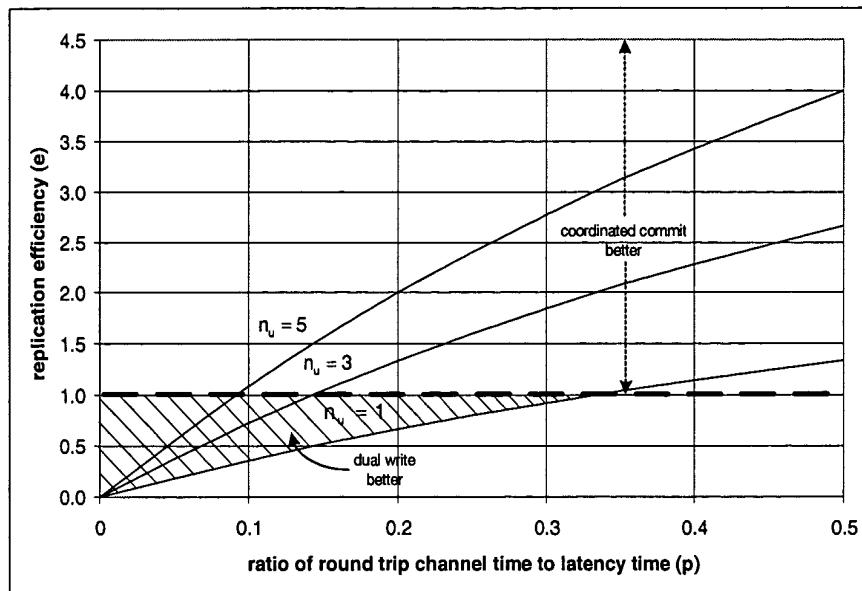
Dual Writes

Fig. 10



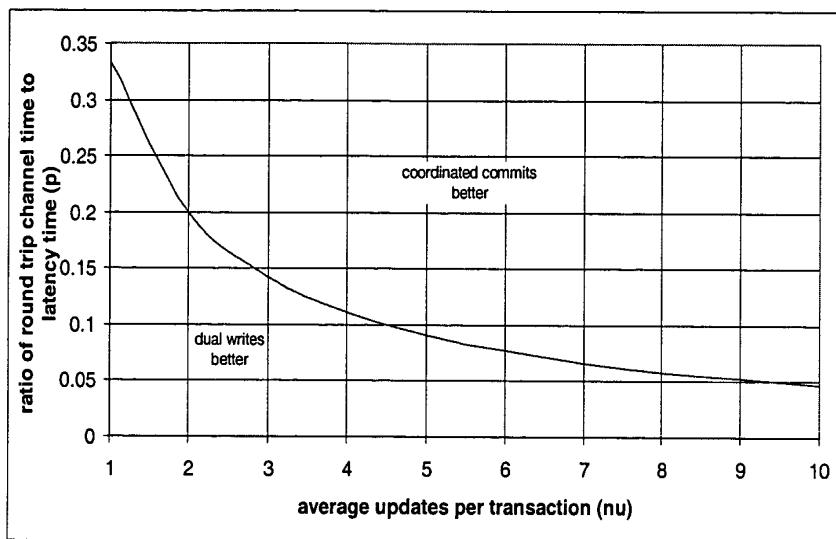
Coordinated Commits

Fig. 11



Synchronous Replication Efficiency

Fig. 12



Equal Efficiency (e=1)

Fig. 13